23.09.06

신서연 김주원

LASSO, RIDGE, AND ELASTIC NET

FOM

Focus On Data Mining

INDEX

01 Introduction

Background

< Erm >

[concept]

Loss

= 손실값 = 각 예시 입력에 대해 **예측값**을 출력할 때 **참인 값과 비교한 잔차**(residual)

Risk

= quantity of expected generalization error

= 도달할 수 있는 **위험한 결과**(의 수치적 양)

" 전체 손실에 대한 평균이 곧 해당 모델의 위험도"

01 Introduction

Background

 # Optimization algorithm

= **최적화** 알고리즘

= **리스크를 최소화** = risk minimization

Empirical risk

= 한정된 훈련데이터에서 경험적으로 추정한 위험데이터 생성 분포

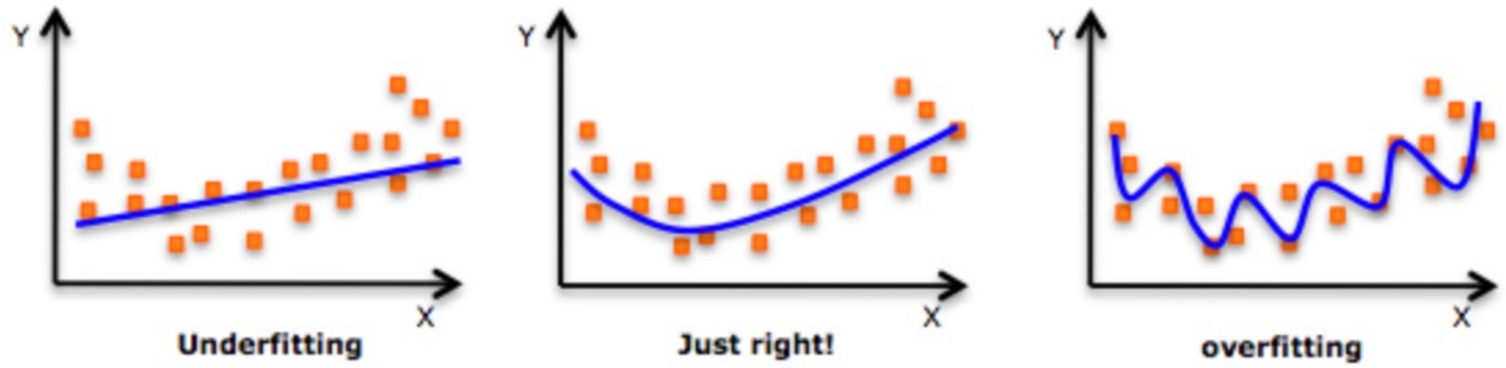
➔ **ERM** = empirical risk minimization

= 한정된 훈련데이터의 분포를 따르는 손실함수의 기댓값을 최소화 시키는 과정

→ ERM 방식에서 오는 문제점 “**overfitting** "

< overfitting >

 # overfitting
= 과적합
= 주어진 데이터에 대해 학습을 계속하여 진행하여 해당 데이터에 **과도하게 편향(biased)된** 상태
= **일반화 성능이 떨어진** 상태



" 모델의 유연성이 과도하게 높을때 / 훈련데이터가 적을때 " 발생

경험적 위험도 = 한정된 훈련 데이터의 분포를 가지고(즉 훈련데이터가 적은 케이스에 해당) 손실함수를 정의
→ overfitting이 발생할 가능성 존재
→ 모델이 학습하는 과정에서 **어떤 규제(패널티)를 주어 과적합을 방지하고자함**. " 규제 = regularization"

<regularization>

= 규제
= 패널티를 주거나 제한을 두어 일반화 능력을 키우는 것

01 Introduction

Background

MLE
= Method of maximum likelihood estimation
= 최대 우도 추정법
= ∅^^=arg max ⟦$p(X|\emptyset)$⟧

p(X|Θ) : 우도
- 사건 X가 일어날 가능성 있는 부류/그룹/집단 Θi에 대한 조건부확률 함수

arg maxΘp(·) : 함수 p(·)를 최대로 만드는, 파라미터(매개변수) Θ의 값
- [참고] ☞ 함수 최대값 파라미터 (arg min, arg max) 참조

^Θ : 우도 Θ를 최대로 하는 추정량
- (추정량 : 관측 표본에 기초하여, 알려지지 않은 모집단 모수를 추정하는 통계량)

결국, 우도 p(X|Θ)를 최대화시키는 파라미터 Θ를 구하려 하는 것. 이것이 최대 우도 추정법

Background

\# NORM
= Vector의 크기(길이)를 측정하는 방법.
- L1 norm = p,q의 각 원소들의 차이의 절대값의 합
- L2 norm = p,q의 유클리디안 거리

\# RSS
= Residual sum of squares
= 잔차 제곱의 합 = sum of squared estimate of error (SSE)
= 통계적 오차의 추정치
= **loss func.(cost func)의 역할!**
= $f(x_i) = \sum_{i=0}^{N} \left(y_i - \sum_{j=0}^{M} x_{ij} W_j \right)^2$

$f(x_i)$ = 추정값
y_i = 관측값
x_i = 입력 데이터의 값
$l(y_n, \emptyset;, x_n)$ = N개의 입력값 x_n에 대하여 각각 y_n을 추정하여 얻은 loss들의 평균값

01 Introduction

Background

L1, L2

L1 regularization , L2 regularization 모두 regularization의 방법 중 하나
ERM에 대해서 알아볼 때에, loss와 loss함수를 잠시 살펴보았음
Loss 함수는 **참데이터와 예측데이터 간 차이를 최소화**하는 값을 구하고자 설계됨

L1 regularization, L2 regularization은 이 **loss 함수의 값** 뿐만 아니라
모델의 복잡도를 최소화하는 **weight값을 더하는 방식으로** 식을 변형한 것.

" Minimize(Loss(Data|Model) + complexity(Model)) " 형태
" L1 regularization ➔ lasso regression, L2 regularization ➔ ridge regression "

complexity(model)
= 모델 복잡도에 대한 계산
= weight값의 최소값을 구하기.
(L1은 weight의 절대값의 최소값을, L2는 weight^2의 값의 합에 대한 최소값을)

01 Introduction

Background

L_2
$= \sqrt{\sum_{i}^{n} x_i^2}$
$= \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$

➔
Ridge regression = $RSS(\beta) + \lambda \sum_{j=1}^{p} \beta_j^2$
$x_n = y_n\ (true) - y_n(predicted)$

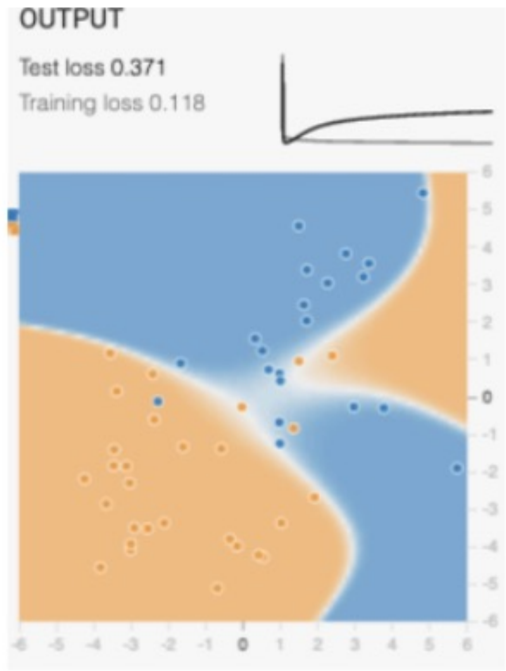
L_1
$= \sum_{i}^{n} |x_i|$
$= |x_1| + |x_2| + |x_3| + \ldots + |x_n|$

➔
Lasso regression = $RSS(\beta) + \sum_{j=1}^{p} |\beta_j|$
$x_n = y_n\ (true) - y_n(predicted)$

01 Introduction

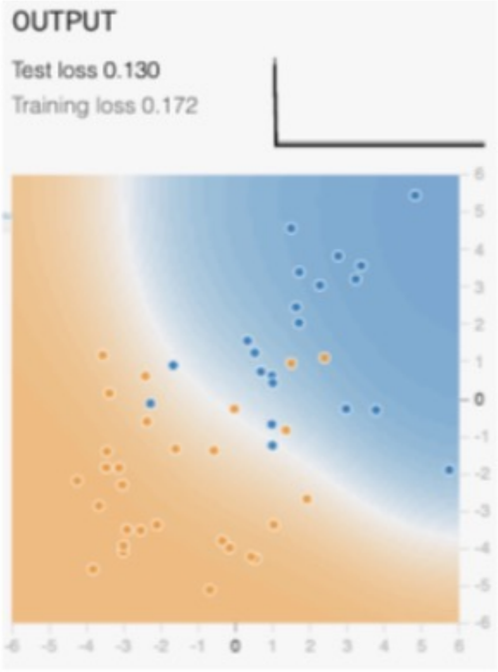
Background

λ = **regularization**의 적용강도 조정하는 하이퍼 파라미터



<without regularization>

<L2 Regularization, lambda=0.03>



<L2 Regularization,lambda=0.3>

lamda = 0 → regularization이 적용되지 않는 상태로, overfitting이 발생한 상태.
lamda = 0.03 → overfitting이 완화된 모습
lamda = 0.3 → overfitting이 거의 발생하지 않고 정상적인 모델이 형성되어 있음.

< Regression >

regression
= 두 변수 x, y 간 관계에 적합한 선

Regression analysis
= 관찰된 연속형 변수들에 대하여 두 변수 사이의 모형을 구한 뒤 적합도를 측정
(= 매개변수 모델을 이용하여 통계적으로 변수들 사이의 관계를 추정
= 입력값(원인)인 독립변수 X의 분포를 분석하여 결과값인 종속변수 Y의 값을 예측)

→ 변수들의 상관성을 표시해줄 수 있는 개념!
+) Linear regression, logistic regression, ridge regression, lasso regression … 등.

→ 회귀분석 식에서 우리가 추정해야 하는 모수가 있을 것인데 이를 추정하는 방법이 OLS

 # OLS
= Ordinary least squares = 오차의 제곱합이 최소가 되는 해를 구하는 방법
= RSS를 최소화 시키는 것

Linear regression and overfitting

- 선형회귀 수식
  - ✓ $y = w_0 + w_1x_1 + w_2x_2$ ....
  - ✓ Train set의 가중치(w_n)와 다르게 작용되는 Test set을 학습시키게 된다면?

⬇

과적합된 모델로 학습하는 양상

02 Relate works

Needs of Constraint

- Linear Regression

✓ 선형 회기에서는 가중치의 비중을 다르게 두어야 하는 Test셋을 학습시킬 때 취약하다

✓ 가중치의 비중을 어떻게 둘 것인가에 따라 다른 유형의 규제(Constraint) 필요

- 모델의 가중치 중 일부를 0으로 만들어 특정 피쳐를 선택하거나 제외
- 모든 가중치를 0에 가깝게 만듬

→ Constraint Regression Model 의 필요

02 | Relate works

Constraint regression models

Constraint regression models

- Ridge Regression
  - L2 규제
- Lasso Regression
  - L1 규제
- Elastic net
  - L1 & L2규제의 혼합

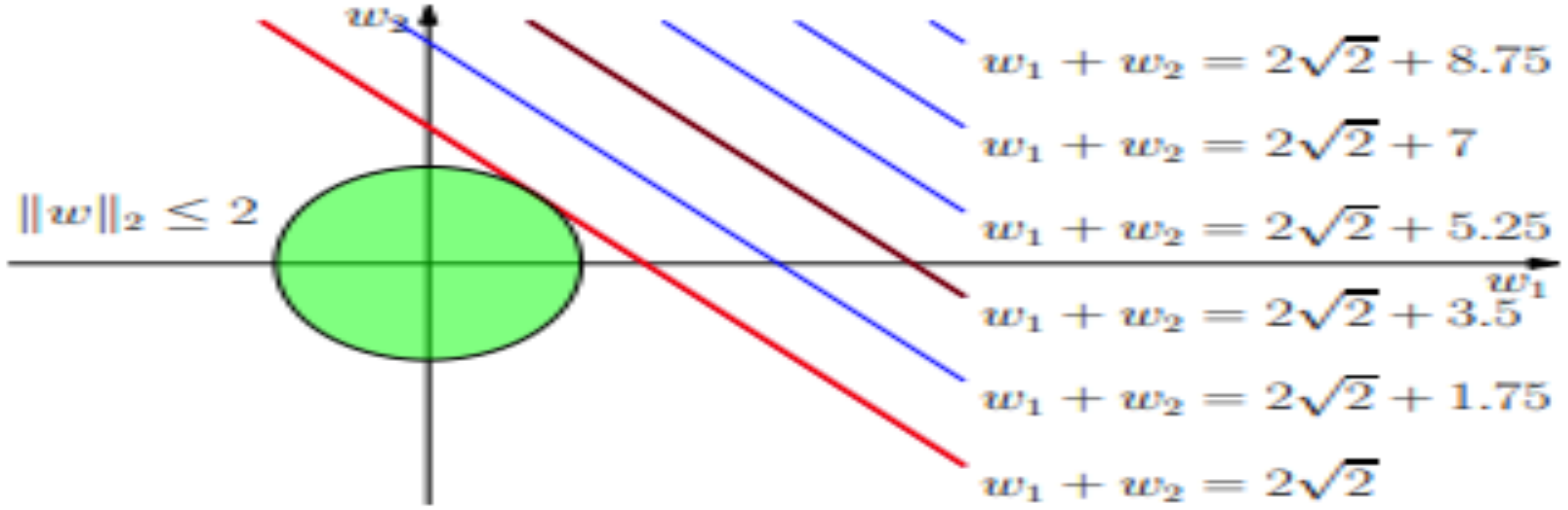
Ridge Regression

- L2 Constraint = $\frac{1}{n}\sum_{i=n}^{N}(y_i - \hat{y})^2 + \lambda\sum_{j=1}^{P}\beta_j^2$

✓ y : 실제값 $\hat{y}$: 예측값

✓ λ : 정규화의 강도를 조절하는 하이퍼 파라미터

✓ β_J : 모델의 가중치



✓ Brown line : ERM

✓ 원 내부는 Loss 가 아니라 Constraint이다 -> Loss를 최소화 하는 최적의 가중치들의 조합!

✓ 수식 : $\|w\|_2 = \sqrt{w_1^2 + w_2^2 + \cdots + w_n^2}$

- w = weight vector
- n = Feature의 수

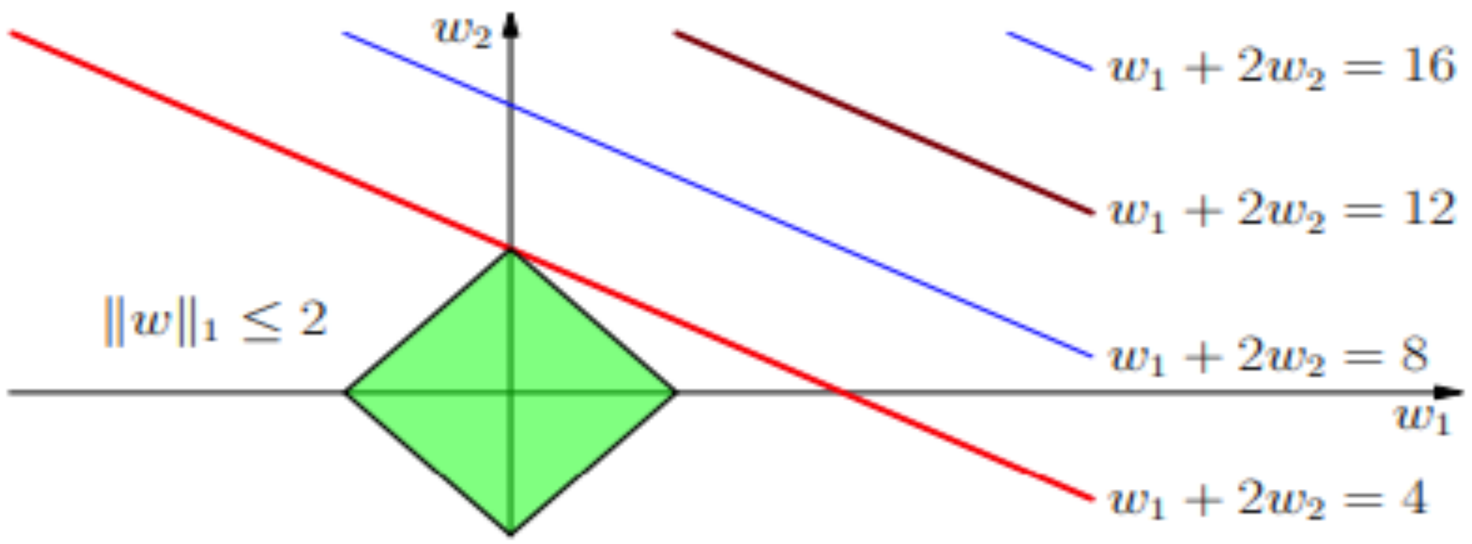
Lasso Regression

- L1 Constraint = $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{P}|\beta_j|$

✓ y : 실제값 $\hat{y}$: 예측값

✓ λ : 정규화의 강도를 조절하는 하이퍼 파라미터

✓ β_J : 모델의 가중치



✓ Brown line : ERM

✓ 마름모 내부는 Loss 가 아니라 Constraint이다 -> Loss를 최소화 하는 최적의 가중치들의 조합!

✓ 수식 : $\|x\| = \sum_{i=1}^{n}|x_i|$

-n : 벡터의 차원 -i : 인덱스(순서)변수

03 Proposed Method

Elastic Net

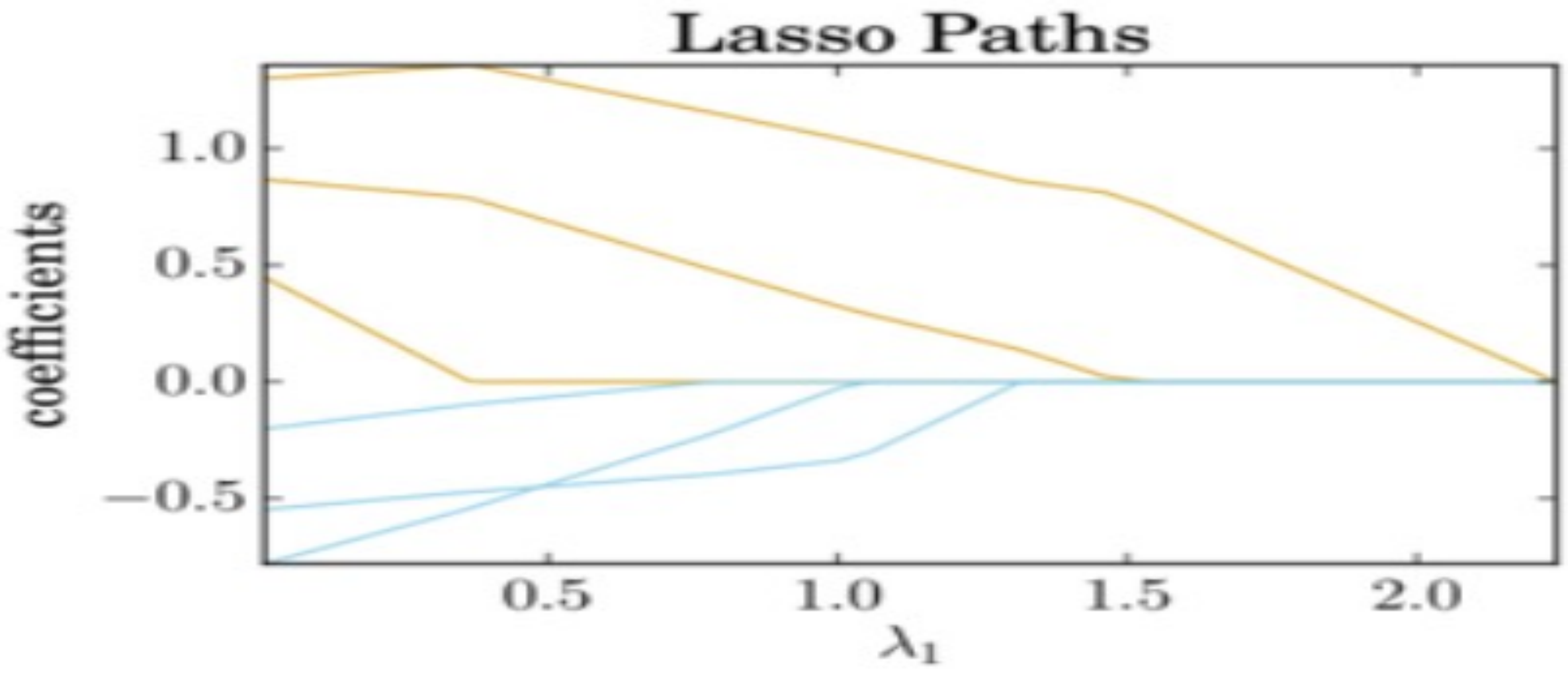
- Elastic Net : $\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)^2 + \lambda_1\|w\|, + \lambda_2\|w\|_2^2$
  - ✓ $\hat{w}$: 목적 함수를 최소화하는 가중치 벡터의 추정치입니다
  - ✓ w : 특성 공간의 차원이 d인 가중치벡터
  - ✓ n : 데이터 셋에 있는 데이터 포인트 수
  - ✓ x : 특성 벡터 y : 타겟 값
  - ✓ $w^T x_i$: 가중치 벡터 w와 특성 벡터 x의 내적, 모델이 y에 대해 예측하는 값
  - ✓ λ_1, λ_2 : 각각 L1, L2노름에 대한 정규화 매개변수
  - ✓ $\|w\|$: l1노름으로 w내 요소들의 절댓값의 합 & l2노름으로 w내의 요소들의 제곱의 합

Lasso, Ridge 정규화 방법의 패널티를 결합한것 -> L1과 L2 정규화 사이의 균형을 찾을 수 있다!

03 Proposed Method

Elastic Net

- Lasso regularization paths:



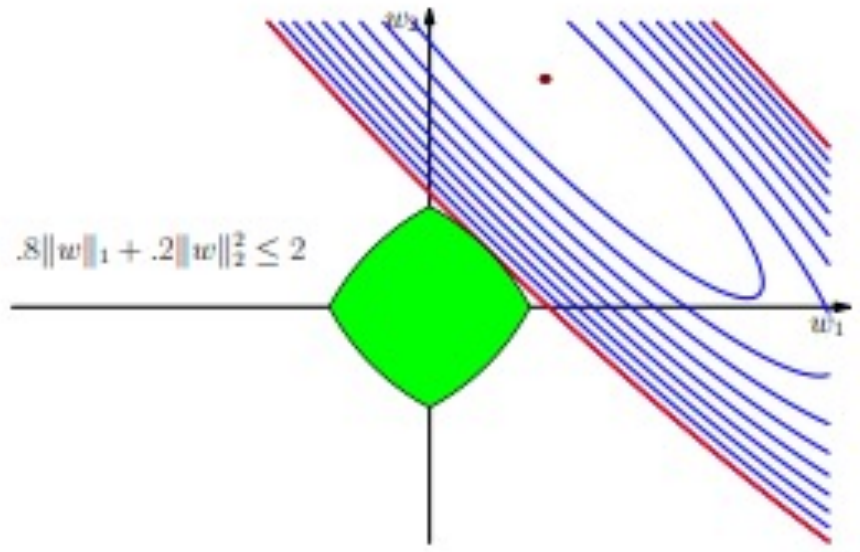
If 노랑선의 변수끼리, 파랑선의 변수끼리는 높은 상관관계가 있다면?

i) lasso 규제에 대한 변수들 마다의 가중치를 나타낸 그래프에서 변수의 다중공선성이 발생할 수 있다

ii) lasso를 사용하면 실제로 영향이 있는 변수가 A인데도 불구하고 A의 가중치를 0으로 만들어 버릴 수 있다

이러한 경우를 예방하기 위해 Elastic net을 사용!

Elastic Net



- Elastic net solution is closer to $w_2 = w_1$ line, despite high correlation.

✓ 상관 관계가 높은 A와 B라는 변수가 있을 때 이 w를 비슷하게 가져가면서 절충안을 찾게됨

04 Experiment

India house price data analysis

• BHK, Size, Floor, Bathroom에 따른 회귀분석을 활용하는 대표적인 데이터 분석인 집값 예측 데이터 분석을 진행

✓ BHK : 침실(Bedroom), 홀(Hall), 주방(Kitchen)의 수

✓ Size : 크기

✓ Floor : 층 수

✓ Bathroom : 욕실의 수

✓ Rent : 임대료

-> 데이터 전처리 과정은 ipynb파일 참고

	BHK	Size	Floor	Bathroom	Rent
0	2	1100	2.0	2	10000
1	2	800	1.0	1	20000
2	2	1000	1.0	1	17000
3	2	800	1.0	1	10000
4	2	850	1.0	1	7500
...	...	...	...	...	...
4741	2	1000	3.0	2	15000
4742	3	2000	1.0	3	29000
4743	3	1750	3.0	3	35000
4744	3	1500	23.0	2	45000
4745	2	1000	4.0	2	15000

4745 rows × 5 columns

Use Linear Regression

• Linear Regression

✓ 선형 회귀 모델 생성 & 훈련 (scikit-learn 사용)

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

```
lr = LinearRegression()
lr
```

```
▾ LinearRegression
LinearRegression()
```

✓ 테스트 데이터로 훈련 & MSE값 확인 ->

```
#테스트 데이터로 훈련
pred = lr.predict(X_test)
```

```
#평균제곱 오차계산(mse)
lin_mse = mean_squared_error(y_test, pred)
print(f"평균 제곱 오차 : {lin_mse}")
```

```
평균 제곱 오차 : 2542273917.5550103
```

MSE : 2542273917.5550103

04 Experiment

Use Ridge Regression

- Ridge Regression
  - ✓ 릿지 회귀 모델 생성 & 훈련 (scikit-learn 사용)

```
from sklearn.linear_model import Ridge
```

```
ridge = Ridge(alpha=1.0)
ridge.fit(X_train, y_train)
```

```
▾ Ridge
Ridge()
```

  - ✓ 테스트 데이터로 훈련 & MSE값 확인

```
Ridge_y_pred = ridge.predict(X_test)
ridge_mse = mean_squared_error(y_test, Ridge_y_pred)
print(f'평균 제곱 오차: {ridge_mse}')
```

```
평균 제곱 오차: 2542185810.6147437
```

Use Lasso Regression

- Lasso Regression

✓ 라쏘 회귀 모델 생성 & 훈련 (scikit-learn 사용)

```
from sklearn.linear_model import Lasso
```

```
lasso = Lasso(alpha=1.0)
lasso.fit(X_train, y_train)
```

```
▾ Lasso
Lasso()
```

✓ 테스트 데이터로 훈련 & MSE값 확인

```
Lasso_y_pred = lasso.predict(X_test)
lasso_mse = mean_squared_error(y_test, Lasso_y_pred)
print(f'평균 제곱 오차: {lasso_mse}')
# 2554775777.0457582
```

```
평균 제곱 오차: 2542242812.453077
```

04 Experiment

Use Elastic Net

- Elastic net
  - ✓ Elastic net 모델 생성 & 학습(scikit-learn 사용)

```
from sklearn.linear_model import ElasticNet
```

```
elastic_net = ElasticNet(alpha=1.0, l1_ratio=0.5)  # alpha는 정규화 매개변수, l1_ratio는 L1 규제의 비율
elastic_net.fit(X_train, y_train)
```

```
▾ ElasticNet
ElasticNet()
```

  - ✓ 테스트 데이터로 훈련 & MSE값 확인

```
Ela_y_pred = elastic_net.predict(X_test)
Ela_mse = mean_squared_error(y_test, Ela_y_pred)
print(f"평균 제곱 오차: {Ela_mse}")
```

```
평균 제곱 오차: 2507599391.550592
```

05 | Conclusion

Ridge Regression VS Lasso Regression VS Elastic Net

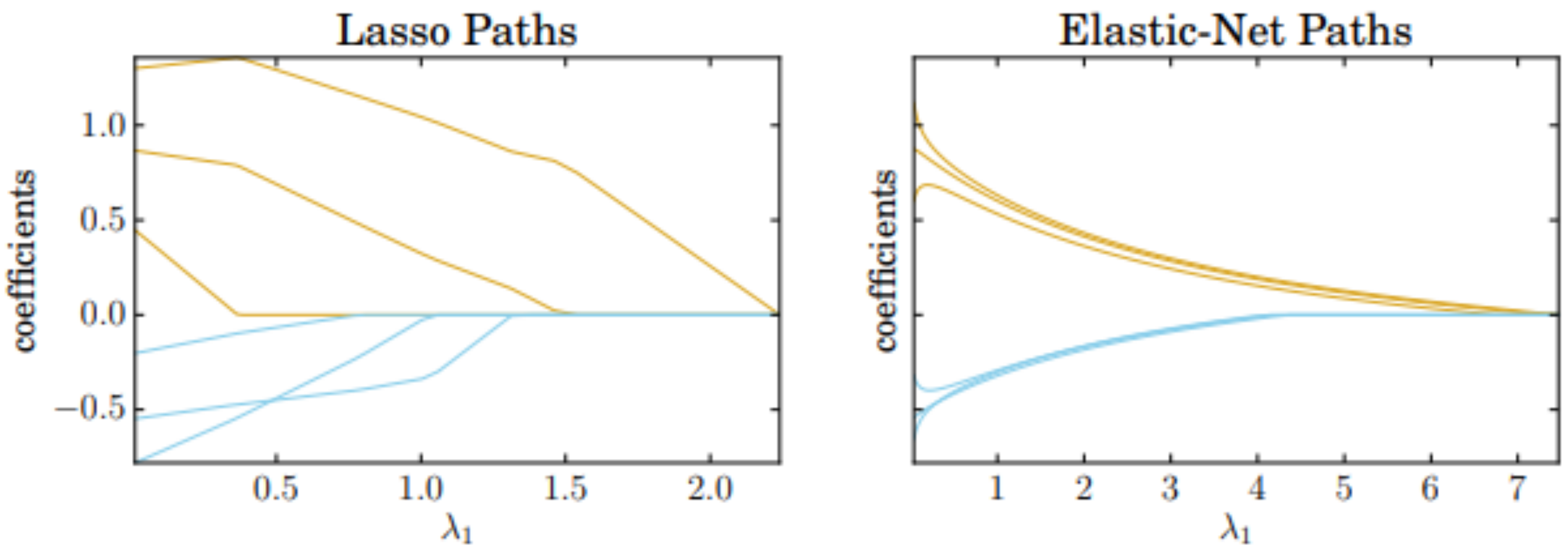
- Elastic Net이 가장 낮은 MSE값을 가진다
  - ✓ L1, L2규제를 모두 활용하여 두 모델의 장점, 단점을 적절히 조합한 결과

- Linear Regression에서 가장 큰 MSE값을 가진다
  - ✓ 선형회귀에서 가장 모델이 데이터를 잘 적합하지 못했음을 확인
  - ✓ 규제가 없는 선형회귀에서 모델이 훈련데이터에 과적합 될 수 있음

- Ridge Regression과 Lasso Regression 에서 비슷한 MSE값을 가진다
  - ✓ 두 모델이 이 데이터셋에 대해 비슷한 복잡도를 가진다
  - ✓ But Ridge Regression에서 약간 더 낮은 MSE값을 가지므로 이 모델에서는 L2규제가 약간 더 효과적일수 있다

05 Conclusion

Difference between Lasso and Elastic Net



- Lasso on left; Elastic net on right.
- Ratio of ℓ_2 to ℓ_1 regularization roughly $2:1$.

- Lasso Regression에 비해 Elastic Net 에서는 weight의 스케일을 비율로써 규제
  - ✓ 값의 극단적인 변화를 부드럽게함과 동시에 최대한 비슷한 가중치를 갖게 끔 절충안을 찾은 모습 확인가능

Q & A

감사합니다